

resource to a privilege. The application identifier is associated with an application program. The system also includes a processor configured to execute computer-executable instructions to determine, responsive to a request from the application program for the resource, the privilege from the manifest stored in the memory area as a function of the application identifier and the resource. The processor is further configured to execute computer-executable instructions to grant the application program access to the resource according to the determined privilege.

[0012] In accordance with another aspect of the invention, a method uninstalls a particular application program from a computing system. The particular application program has at least one file associated therewith. The particular application program is one of a plurality of application programs installed on the computing system. The method includes receiving a request to uninstall the particular application program. The method also includes determining an identifier associated with the particular application program. The method further includes identifying, via the determined identifier, a file associated only with the particular application program of the plurality of application programs. The identified file has the determined identifier associated therewith. The method also includes deleting the identified file.

[0013] Alternatively, the invention may comprise various other methods and apparatuses.

[0014] Other features will be in part apparent and in part pointed out hereinafter.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] **FIG. 1** is an exemplary embodiment of an operating system providing an application program with access to a resource.

[0016] **FIG. 2** is an exemplary flow chart illustrating operation of an access control method.

[0017] **FIG. 3** is an exemplary flow chart illustrating a mitigation architecture for protecting various resources.

[0018] **FIG. 4** is an exemplary flow chart illustrating operation of a method of providing access control for files, system settings, and extensions.

[0019] **FIG. 5** is an exemplary flow chart illustrating operation of a method of providing access control for system settings.

[0020] **FIG. 6** is an exemplary flow chart illustrating operation of the removal of an installed application program from a computing system.

[0021] **FIG. 7** is a block diagram illustrating one example of a suitable computing system environment in which the invention may be implemented.

[0022] Corresponding reference characters indicate corresponding parts throughout the drawings.

DETAILED DESCRIPTION OF THE INVENTION

[0023] In one embodiment, the invention provides a method for protecting resources. In particular, functionality of the operating system enables the declaration of protection for files and system settings. The declared protection is

persisted and enforced by the operating system or other application program through a set of actions the operating system may use during the application lifecycle to manage, track, predict, and mitigate the installation, running, servicing, and removal of application programs. Resource protection provides referential integrity of the vital system data (e.g., file associations), addresses application fragility issues to improve reliability and consistency by tracing and isolating access to resources by each application program, and manages the impact of interactions by the system and applications with protected resources. For example, embodiments of the invention may be used to provide security against applications that have been infected by a virus or a worm. Embodiments of the invention are operable with any operating system model to provide extensibility and enable integration. The resource protection strategies and implementation of embodiments of the invention also prevent an application installer from accidentally or maliciously modifying or replacing vital system resources. Embodiments of the invention may be combined with other strategies for protecting system resources. For example, a computing system may implement strategies including a combination of lock down, isolation, virtualization, transaction, and sandboxing.

[0024] Referring first to **FIG. 1**, an exemplary embodiment of an operating system **102** provides an application program **104** with access to a resource per a manifest **108**. A resource includes, but is not limited to, a file, folder, process, thread, system setting, named object, an application programming interface (API), a specific code path, a library of executable routines, operating system property value, and an operating system resource. A named object includes any object identified by alphabetic, numeric, alphanumeric, or non-human readable (e.g., a globally unique identifier) data. For example, any object that may be protected by a user identity may be protected by an application identity (e.g., a network socket). For example, a number of APIs and code paths provide send mail capability, and access to these APIs and code paths might be restricted. In another example, the ability to reboot the system is restricted. Resources also include the system's name space (e.g., the 'names' themselves), not just specific named objects. For example, reserving or 'squatting' on a name before an object is created with the name creates both fragility and security issues.

[0025] In one embodiment, the manifest such as manifest **108** presented by the application program (e.g., application program **104**) indicates the privileges that the application program would like to have. In operation, the operating system may grant or deny some of the requested privileges resulting in a computed or effective manifest that the operating system maintains for the application program.

[0026] Each application program such as application program **104** is assigned an application identifier to distinguish the application program from other application programs. In one embodiment, the application identifier is assigned to a group of application programs to enable each application program in the group of application programs to have the same access or privilege to resources as the other application programs in the group. In **FIG. 1**, the application program **104** has identifier **ID1** associated therewith. The operating system **102** intercepts an attempt by the application program **104** to access a resource such as Resource A **106**. The operating system **102** consults the manifest **108** to determine